# Exercise 1: Implementing the Singleton Pattern (C#)

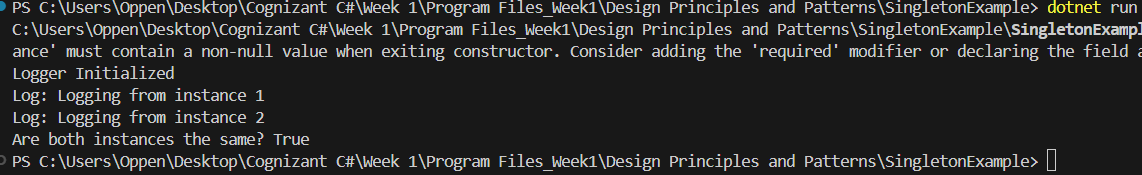
Design Principles and Patterns

Scenario:  
You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

## Code: SingletonExample.cs

using System;  
  
public class Logger  
{  
 private static Logger instance;  
 private Logger()  
 {  
 Console.WriteLine("Logger Initialized");  
 }  
  
 public static Logger GetInstance()  
 {  
 if (instance == null)  
 {  
 instance = new Logger();  
 }  
 return instance;  
 }  
  
 public void Log(string message)  
 {  
 Console.WriteLine("Log: " + message);  
 }  
}  
  
public class SingletonExample  
{  
 public static void Main(string[] args)  
 {  
 Logger logger1 = Logger.GetInstance();  
 logger1.Log("Logging from instance 1");  
  
 Logger logger2 = Logger.GetInstance();  
 logger2.Log("Logging from instance 2");  
  
 Console.WriteLine("Are both instances the same? " + (logger1 == logger2));  
 }  
}

## Output Screenshot



# Exercise 2: Implementing the Factory Method Pattern (C#)

Scenario:  
You are developing a document management system that needs to create different types of documents (e.g., Word, PDF, Excel). Use the Factory Method Pattern to achieve this.

## Code: FactoryPatternExample.cs

using System;  
  
public interface IDocument  
{  
 void Open();  
}  
  
public class WordDocument : IDocument  
{  
 public void Open() => Console.WriteLine("Opening Word document.");  
}  
  
public class PdfDocument : IDocument  
{  
 public void Open() => Console.WriteLine("Opening PDF document.");  
}  
  
public class ExcelDocument : IDocument  
{  
 public void Open() => Console.WriteLine("Opening Excel document.");  
}  
  
public abstract class DocumentFactory  
{  
 public abstract IDocument CreateDocument();  
}  
  
public class WordDocumentFactory : DocumentFactory  
{  
 public override IDocument CreateDocument() => new WordDocument();  
}  
  
public class PdfDocumentFactory : DocumentFactory  
{  
 public override IDocument CreateDocument() => new PdfDocument();  
}  
  
public class ExcelDocumentFactory : DocumentFactory  
{  
 public override IDocument CreateDocument() => new ExcelDocument();  
}  
  
public class FactoryPatternExample  
{  
 public static void Main(string[] args)  
 {  
 DocumentFactory wordFactory = new WordDocumentFactory();  
 IDocument word = wordFactory.CreateDocument();  
 word.Open();  
  
 DocumentFactory pdfFactory = new PdfDocumentFactory();  
 IDocument pdf = pdfFactory.CreateDocument();  
 pdf.Open();  
  
 DocumentFactory excelFactory = new ExcelDocumentFactory();  
 IDocument excel = excelFactory.CreateDocument();  
 excel.Open();  
 }  
}

## Output Screenshot

